**TARGET SQL**

1. **In the database given, we have 6 tables. Please find exploratory analysis steps like checking the structure & characteristics of the dataset for the tables given below:**

* Customers table: There are a total of 99441 rows. There were no missing values found in any of the features.

|  |  |
| --- | --- |
| Column | Data type |
| customer\_id | STRING |
| customer\_unique\_id | STRING |
| customer\_zip\_code\_prefix | INTEGER |
| customer\_city | STRING |
| customer\_state | STRING |

There are 4119 distinct cities and 27 distinct states in our dataset.

* Geolocation table: There are a total of 1,000,163 rows. There were no missing values found in any of the features.

|  |  |
| --- | --- |
| Column | Data type |
| geolocation\_zip\_code\_prefix | INTEGER |
| geolocation\_lat | FLOAT |
| geolocation\_lng | FLOAT |
| geolocation\_city | STRING |
| geolocation\_state | STRING |

* order\_items table: There are a total of 112,650 rows. There were no missing values found in any of the features.

|  |  |
| --- | --- |
| Column | Data type |
| order\_id | STRING |
| order\_item\_id | INTEGER |
| product\_id | STRING |
| seller\_id | STRING |
| shipping\_limit\_date | TIMESTAMP |
| price | FLOAT |
| freight\_value | FLOAT |

* order\_reviews table: There are a total of 99224 rows. I found 87675 Null values in review\_comment\_title feature. Reviews were created and answered from 31-12-2017 to 01-01-2018.

|  |  |
| --- | --- |
| Column | Data type |
| review\_id | STRING |
| order\_id | STRING |
| review\_score | INTEGER |
| review\_comment\_title | STRING |
| review\_creation\_date | TIMESTAMP |
| review\_answer\_timestamp | TIMESTAMP |

* orders table: There are a total of 99441 rows. I found 160 Null values in order\_approved\_at**,** 1783 Null values in order\_delivered\_carrier\_date and 2965 NULL values in order\_delivered\_customer\_date columns. We have orders purchased from 4th Sep 2016 to 17th Oct 2018.

|  |  |
| --- | --- |
| Column | Data type |
| order\_id | STRING |
| customer\_id | STRING |
| order\_status | STRING |
| order\_purchase\_timestamp | TIMESTAMP |
| order\_approved\_at | TIMESTAMP |
| order\_delivered\_carrier\_date | TIMESTAMP |
| order\_delivered\_customer\_date | TIMESTAMP |
| order\_estimated\_delivery\_date | TIMESTAMP |

* Payments table: There are a total of 103886 rows. There were no missing values found in any of the features.

|  |  |
| --- | --- |
| Column | Data type |
| order\_id | STRING |
| payment\_sequential | INTEGER |
| payment\_type | STRING |
| payment\_installments | INTEGER |
| payment\_value | FLOAT |

* products table: There are a total of 32951 rows. I found 610 Null values in product\_category**,** product\_name\_length, product\_description\_length and product\_photos\_qty columns. There were 2 null values in the product weight and dimension columns.

|  |  |
| --- | --- |
| Column | Data type |
| product\_id | STRING |
| product\_category | STRING |
| product\_name\_length | INTEGER |
| product\_description\_length | INTEGER |
| product\_photos\_qty | INTEGER |
| product\_weight\_g | INTEGER |
| product\_length\_cm | INTEGER |
| product\_height\_cm | INTEGER |
| product\_width\_cm | INTEGER |

* sellers table: There are a total of 3095 rows. There were no missing values found in any of the features.

|  |  |
| --- | --- |
| Column | Data type |
| seller\_id | STRING |
| seller\_zip\_code\_prefix | INTEGER |
| seller\_city | STRING |
| seller\_state | STRING |

1. **How many orders do we have for each order status?**

Query:

SELECT order\_status, COUNT(\*) as total\_count

FROM `scalerproject4.targetsql.orders`

GROUP BY order\_status

Answer:

|  |  |
| --- | --- |
| order\_status | total\_count |
| created | 5 |
| shipped | 1107 |
| approved | 2 |
| canceled | 625 |
| invoiced | 314 |
| delivered | 96478 |
| processing | 301 |
| unavailable | 609 |

1. **Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario?**

Query:

* SELECT years, months, COUNT(\*) as month\_count

FROM

(

SELECT EXTRACT(MONTH FROM order\_purchase\_timestamp) as months, EXTRACT(YEAR FROM order\_purchase\_timestamp) as years

FROM `scalerproject4.targetsql.orders`

)

GROUP BY years, months

ORDER BY years, months

* SELECT years, COUNT(\*) as year\_count

FROM

(

SELECT EXTRACT(YEAR FROM order\_purchase\_timestamp) as years

FROM `scalerproject4.targetsql.orders`

)

GROUP BY years

ORDER BY years

Answer:

We can see that the no of orders at Target are increasing every year and every month. This clearly shows the growth in ecommerce.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **years** | **months** | **month\_count** | **years** | **year\_count** |
| 2016 | 9 | 4 | 2016 | 329 |
| 2016 | 10 | 324 | 2017 | 45101 |
| 2016 | 12 | 1 | 2018 | 54011 |
| 2017 | 1 | 800 |
| 2017 | 2 | 1780 |
| 2017 | 3 | 2682 |
| 2017 | 4 | 2404 |
| 2017 | 5 | 3700 |
| 2017 | 6 | 3245 |
| 2017 | 7 | 4026 |
| 2017 | 8 | 4331 |
| 2017 | 9 | 4285 |
| 2017 | 10 | 4631 |
| 2017 | 11 | 7544 |
| 2017 | 12 | 5673 |
| 2018 | 1 | 7269 |
| 2018 | 2 | 6728 |
| 2018 | 3 | 7211 |
| 2018 | 4 | 6939 |
| 2018 | 5 | 6873 |
| 2018 | 6 | 6167 |
| 2018 | 7 | 6292 |
| 2018 | 8 | 6512 |
| 2018 | 9 | 16 |
| 2018 | 10 | 4 |

1. **On what day of week brazilians customers tend to do online purchasing?**

Query:

SELECT \*,

  CASE

  WHEN day\_of\_week = 1

  THEN 'SUNDAY'

  WHEN day\_of\_week = 2

  THEN 'MONDAY'

  WHEN day\_of\_week = 3

  THEN 'TUESDAY'

  WHEN day\_of\_week = 4

  THEN 'WEDNESDAY'

  WHEN day\_of\_week = 5

  THEN 'THURSDAY'

  WHEN day\_of\_week = 6

  THEN 'FRIDAY'

  WHEN day\_of\_week = 7

  THEN 'SATURDAY'

  END AS day\_of\_week\_names

FROM

(

SELECT EXTRACT(DAYOFWEEK FROM order\_purchase\_timestamp) as day\_of\_week, count(\*) AS week\_day\_count

FROM `scalerproject4.targetsql.orders`

GROUP BY day\_of\_week

ORDER BY week\_day\_count DESC

)

|  |  |  |
| --- | --- | --- |
| day\_of\_week | week\_day\_count | day\_of\_week\_names |
| 2 | 16196 | Monday |
| 3 | 15963 | Tuesday |
| 4 | 15552 | Wednesday |
| 5 | 14761 | Thursday |
| 6 | 14122 | Friday |
| 1 | 11960 | Sunday |
| 7 | 10887 | Saturday |

Answer: We can see that most number of purchases are made on weekdays as compared to weekends with Monday topping the table.

1. **What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?**

Query:

SELECT \*, count(\*) AS tot\_order\_count

FROM

(

SELECT

  Case

  WHEN EXTRACT(HOUR FROM order\_purchase\_timestamp) IN (23, 0, 1,2,3,4)

  THEN 'NIGHT : 23:00 to 04:59'

  WHEN EXTRACT(HOUR FROM order\_purchase\_timestamp) IN (5,6,7,8,9,10,11)

  THEN 'MORNING : 05:00 to 11:59'

  WHEN EXTRACT(HOUR FROM order\_purchase\_timestamp) IN (12,13,14,15,16)

  THEN 'AFTERNOON : 12:00 to 16:59'

  WHEN EXTRACT(HOUR FROM order\_purchase\_timestamp) IN (17,18,19,20,21,22)

  THEN 'EVENING : 17:00 to 22:59'

  END AS times\_of\_day

FROM `scalerproject4.targetsql.orders`

)

GROUP BY times\_of\_day

ORDER BY tot\_order\_count DESC

|  |  |
| --- | --- |
| **times\_of\_day** | **tot\_order\_count** |
| EVENING : 17:00 to 22:59 | 36127 |
| AFTERNOON : 12:00 to 16:59 | 32211 |
| MORNING : 02:00 to 11:59 | 22428 |
| NIGHT : 23:00 to 01:59 | 8675 |

Answer: We can see that most number of purchases are made during the evenings which is from 17:00 to 22:59.

1. **Feature Extraction:** 
   * Order\_purchase\_year

SELECT \*, EXTRACT(YEAR FROM order\_purchase\_timestamp) as year

FROM `scalerproject4.targetsql.orders`

* + Order\_purchase\_month

SELECT \*, EXTRACT(MONTH FROM order\_purchase\_timestamp) as month

FROM `scalerproject4.targetsql.orders`

* + Order\_purchase\_date

SELECT \*, EXTRACT(DATE FROM order\_purchase\_timestamp) as date

FROM `scalerproject4.targetsql.orders`

* + Order\_purchase\_day

SELECT \*, EXTRACT(DAY FROM order\_purchase\_timestamp) as day

FROM `scalerproject4.targetsql.orders`

* + Order\_purchase\_dayofweek

SELECT \*, EXTRACT(DAYOFWEEK FROM order\_purchase\_timestamp) as day\_of\_week

FROM `scalerproject4.targetsql.orders`

* + Order\_purchase\_dayofweek\_name

SELECT \*,

  CASE

  WHEN day\_of\_week = 1

  THEN 'SUNDAY'

  WHEN day\_of\_week = 2

  THEN 'MONDAY'

  WHEN day\_of\_week = 3

  THEN 'TUESDAY'

  WHEN day\_of\_week = 4

  THEN 'WEDNESDAY'

  WHEN day\_of\_week = 5

  THEN 'THURSDAY'

  WHEN day\_of\_week = 6

  THEN 'FRIDAY'

  WHEN day\_of\_week = 7

  THEN 'SATURDAY'

  END AS day\_of\_week\_names

FROM

(

SELECT \*, EXTRACT(DAYOFWEEK FROM order\_purchase\_timestamp) as day\_of\_week

FROM `scalerproject4.targetsql.orders`

)

* + Order\_purchase\_hour

SELECT \*, EXTRACT(HOUR FROM order\_purchase\_timestamp) as hour

FROM `scalerproject4.targetsql.orders`

* + Order\_purchase\_time\_day

SELECT \*, EXTRACT(TIME FROM order\_purchase\_timestamp) as time

FROM `scalerproject4.targetsql.orders`

1. **Month on month orders by region**

Query:

SELECT cu.customer\_state, EXTRACT(YEAR FROM order\_purchase\_timestamp) as year, EXTRACT(MONTH FROM order\_purchase\_timestamp) as month, count(\*) as state\_count

FROM `scalerproject4.targetsql.orders` AS ord

LEFT JOIN `scalerproject4.targetsql.Customers` as cu

  ON cu.customer\_id = ord.customer\_id

GROUP BY cu.customer\_state, year, month

ORDER BY cu.customer\_state, year, month

LIMIT 10

Answer:

|  |  |  |  |
| --- | --- | --- | --- |
| customer\_state | year | month | state\_count |
| AC | 2017 | 1 | 2 |
| AC | 2017 | 2 | 3 |
| AC | 2017 | 3 | 2 |
| AC | 2017 | 4 | 5 |
| AC | 2017 | 5 | 8 |
| AC | 2017 | 6 | 4 |
| AC | 2017 | 7 | 5 |
| AC | 2017 | 8 | 4 |
| AC | 2017 | 9 | 5 |
| AC | 2017 | 10 | 6 |

1. **Total of customer orders by state**

**Query:**

SELECT cu.customer\_state, count(\*) as state\_count

FROM `scalerproject4.targetsql.orders` AS ord

LEFT JOIN `scalerproject4.targetsql.Customers` as cu

  ON cu.customer\_id = ord.customer\_id

GROUP BY cu.customer\_state

ORDER BY state\_count DESC

Answer:

|  |  |
| --- | --- |
| customer\_state | state\_count |
| SP | 41746 |
| RJ | 12852 |
| MG | 11635 |
| RS | 5466 |
| PR | 5045 |
| SC | 3637 |
| BA | 3380 |
| DF | 2140 |
| ES | 2033 |
| GO | 2020 |
| PE | 1652 |
| CE | 1336 |
| PA | 975 |
| MT | 907 |
| MA | 747 |
| MS | 715 |
| PB | 536 |
| PI | 495 |
| RN | 485 |
| AL | 413 |
| SE | 350 |
| TO | 280 |
| RO | 253 |
| AM | 148 |
| AC | 81 |
| AP | 68 |
| RR | 46 |

1. **Top 10 Brazilian cities most no. of orders**

Query:

SELECT cu.customer\_city, count(\*) as city\_count

FROM `scalerproject4.targetsql.orders` AS ord

LEFT JOIN `scalerproject4.targetsql.Customers` as cu

  ON cu.customer\_id = ord.customer\_id

GROUP BY cu.customer\_city

ORDER BY city\_count DESC

LIMIT 10

Answer:

|  |  |
| --- | --- |
| customer\_city | city\_count |
| sao paulo | 15540 |
| rio de janeiro | 6882 |
| belo horizonte | 2773 |
| brasilia | 2131 |
| curitiba | 1521 |
| campinas | 1444 |
| porto alegre | 1379 |
| salvador | 1245 |
| guarulhos | 1189 |
| sao bernardo do campo | 938 |

1. **How are customers distributed in Brazil**

Query:

SELECT customer\_state, count(\*) as state\_customer\_count

FROM `scalerproject4.targetsql.Customers`

GROUP BY customer\_state

ORDER BY state\_customer\_count DESC

Answer: Highest customer count in SP state.

|  |  |
| --- | --- |
| customer\_state | state\_customer\_count |
| SP | 41746 |
| RJ | 12852 |
| MG | 11635 |
| RS | 5466 |
| PR | 5045 |
| SC | 3637 |
| BA | 3380 |
| DF | 2140 |
| ES | 2033 |
| GO | 2020 |
| PE | 1652 |
| CE | 1336 |
| PA | 975 |
| MT | 907 |
| MA | 747 |
| MS | 715 |
| PB | 536 |
| PI | 495 |
| RN | 485 |
| AL | 413 |
| SE | 350 |
| TO | 280 |
| RO | 253 |
| AM | 148 |
| AC | 81 |
| AP | 68 |
| RR | 46 |

1. **City wise number of unique customers**

Query:

SELECT customer\_city, count(DISTINCT customer\_id) as city\_customer\_count

FROM `scalerproject4.targetsql.Customers`

GROUP BY customer\_city

ORDER BY city\_customer\_count DESC

LIMIT 10

Answer: Top count in Sao Paulo city.

|  |  |
| --- | --- |
| customer\_city | city\_customer\_count |
| sao paulo | 15540 |
| rio de janeiro | 6882 |
| belo horizonte | 2773 |
| brasilia | 2131 |
| curitiba | 1521 |
| campinas | 1444 |
| porto alegre | 1379 |
| salvador | 1245 |
| guarulhos | 1189 |
| sao bernardo do campo | 938 |

1. **Impact on Economy: Analyze the money movemented by e-commerce by looking at order prices, freight and others.**

Queries:

* + CTE:

WITH orders\_cte AS

(

  SELECT EXTRACT(MONTH FROM od.order\_purchase\_timestamp) as purchase\_month,

    EXTRACT(YEAR FROM od.order\_purchase\_timestamp) as purchase\_year,

    count(od.order\_id) as order\_count, sum(oi.price) as sum\_price, sum(oi.freight\_value) as sum\_freight,

    sum(oi.price) / count(od.order\_id) AS price\_per\_order,

    sum(oi.freight\_value) / count(od.order\_id) AS freight\_per\_order

  FROM `scalerproject4.targetsql.orders` AS od

  LEFT JOIN `scalerproject4.targetsql.order\_items` as oi

    ON od.order\_id = oi.order\_id

  GROUP BY purchase\_month, purchase\_year

)

* + Total amount sold in 2017 between Jan to August:

SELECT SUM(sum\_price)

FROM orders\_cte

WHERE purchase\_year = 2017 AND purchase\_month BETWEEN 1 AND 8

ANS: 3113000.319999788

* + Total amount sold in 2018 between Jan to august:

SELECT SUM(sum\_price)

FROM orders\_cte

WHERE purchase\_year = 2018 AND purchase\_month BETWEEN 1 AND 8

ANS: 7385905.8000043072

* + % increase from 2017 to 2018:

(7385905.8000043072 - 3113000.319999788)/ 3113000.319999788 \* 100

ANS : 137.26 %

* + NEW\_CTE:

WITH orderswithcustomers\_cte AS

(

  SELECT cu.customer\_state, oi.price, oi.freight\_value

  FROM `scalerproject4.targetsql.orders` AS od

  LEFT JOIN `scalerproject4.targetsql.order\_items` as oi

    ON od.order\_id = oi.order\_id

  LEFT JOIN `scalerproject4.targetsql.Customers` as cu

    ON cu.customer\_id = od.customer\_id

)

* + Mean & Sum of price by customer state:

SELECT customer\_state, SUM(price) as price\_sum, AVG(price) as price\_avg

FROM orderswithcustomers\_cte

GROUP BY customer\_state

ORDER BY customer\_state

|  |  |  |
| --- | --- | --- |
| customer\_state | price\_sum | price\_avg |
| AC | 15982.95 | 173.7277174 |
| AL | 80314.81 | 180.8892117 |
| AM | 22356.84 | 135.496 |
| AP | 13474.3 | 164.3207317 |
| BA | 511349.99 | 134.6012082 |
| CE | 227254.71 | 153.7582612 |
| DF | 302603.94 | 125.7705486 |
| ES | 275037.31 | 121.9137012 |
| GO | 294591.95 | 126.2717317 |
| MA | 119648.22 | 145.2041505 |
| MG | 1585308.03 | 120.7485741 |
| MS | 116812.64 | 142.6283761 |
| MT | 156453.53 | 148.2971848 |
| PA | 178947.81 | 165.6924167 |
| PB | 115268.08 | 191.4752159 |
| PE | 262788.03 | 145.5083223 |
| PI | 86914.08 | 160.3580812 |
| PR | 683083.76 | 119.0041394 |
| RJ | 1824092.67 | 125.1178181 |
| RN | 83034.98 | 156.9659357 |
| RO | 46140.64 | 165.9735252 |
| RR | 7829.43 | 150.5659615 |
| RS | 750304.02 | 120.3374531 |
| SC | 520553.34 | 124.6535776 |
| SE | 58920.85 | 153.0411688 |
| SP | 5202955.05 | 109.6536292 |
| TO | 49621.74 | 157.5293333 |

* + Mean & Sum of freight value by customer state:

SELECT customer\_state, SUM(freight\_value) as freight\_value\_sum, AVG(freight\_value) as freight\_value\_avg

FROM orderswithcustomers\_cte

GROUP BY customer\_state

ORDER BY customer\_state

|  |  |  |
| --- | --- | --- |
| customer\_state | freight\_value\_sum | freight\_value\_avg |
| AC | 3686.75 | 40.07336957 |
| AL | 15914.59 | 35.84367117 |
| AM | 5478.89 | 33.20539394 |
| AP | 2788.5 | 34.00609756 |
| BA | 100156.68 | 26.36395894 |
| CE | 48351.59 | 32.71420162 |
| DF | 50625.5 | 21.04135495 |
| ES | 49764.6 | 22.0587766 |
| GO | 53114.98 | 22.76681526 |
| MA | 31523.77 | 38.25700243 |
| MG | 270853.46 | 20.63016681 |
| MS | 19144.03 | 23.374884 |
| MT | 29715.43 | 28.16628436 |
| PA | 38699.3 | 35.83268519 |
| PB | 25719.73 | 42.72380399 |
| PE | 59449.66 | 32.91786268 |
| PI | 21218.2 | 39.14797048 |
| PR | 117851.68 | 20.53165157 |
| RJ | 305589.31 | 20.96092393 |
| RN | 18860.1 | 35.65236295 |
| RO | 11417.38 | 41.06971223 |
| RR | 2235.19 | 42.98442308 |
| RS | 135522.74 | 21.73580433 |
| SC | 89660.26 | 21.47036877 |
| SE | 14111.47 | 36.65316883 |
| SP | 718723.07 | 15.14727539 |
| TO | 11732.68 | 37.24660317 |

1. **Analysis on sales, freight and delivery time**

Queries:

CTE:

WITH CTE AS

(

SELECT customer\_state, AVG(freight\_value) AS avg\_freight\_value, AVG(time\_to\_delivery) AS avg\_time\_to\_delivery, AVG(diff\_estimated\_delivery) AS avg\_diff\_estimated\_delivery

FROM

(

SELECT cu.customer\_state, oi.freight\_value, DATE\_DIFF(EXTRACT (DATE FROM od.order\_purchase\_timestamp),EXTRACT(DATE FROM od.order\_delivered\_customer\_date), DAY) AS time\_to\_delivery,

  DATE\_DIFF(EXTRACT (DATE FROM od.order\_estimated\_delivery\_date),EXTRACT(DATE FROM od.order\_delivered\_customer\_date), DAY) AS diff\_estimated\_delivery

FROM `scalerproject4.targetsql.orders` as od

LEFT JOIN `scalerproject4.targetsql.Customers` AS cu

  ON cu.customer\_id = od.customer\_id

LEFT JOIN `scalerproject4.targetsql.order\_items` AS oi

  ON oi.order\_id = od.order\_id

)

GROUP BY customer\_state

)

* + Top 5 states with highest/lowest average freight value

Highest:

SELECT \*

FROM  CTE

ORDER BY avg\_freight\_value DESC

LIMIT 5

|  |  |  |  |
| --- | --- | --- | --- |
| customer\_state | avg\_freight\_value | avg\_time\_to\_delivery | avg\_diff\_estimated\_delivery |
| RR | 42.98442308 | -28.17391304 | 18.32608696 |
| PB | 42.72380399 | -20.54607509 | 13.03754266 |
| RO | 41.06971223 | -19.65567766 | 20.04029304 |
| AC | 40.07336957 | -20.68131868 | 20.97802198 |
| PI | 39.14797048 | -19.31739962 | 11.52772467 |

Lowest:

SELECT \*

FROM CTE

ORDER BY avg\_freight\_value ASC

LIMIT 5

|  |  |  |  |
| --- | --- | --- | --- |
| customer\_state | avg\_freight\_value | avg\_time\_to\_delivery | avg\_diff\_estimated\_delivery |
| SP | 15.14727539 | -8.662252654 | 11.20791077 |
| PR | 20.53165157 | -11.89307842 | 13.48610374 |
| MG | 20.63016681 | -11.92072463 | 13.34264922 |
| RJ | 20.96092393 | -15.07479146 | 12.01477449 |
| DF | 21.04135495 | -12.89384289 | 12.20042463 |

* + Top 5 states with highest/lowest average time to delivery

Highest:

SELECT \*

FROM CTE

ORDER BY avg\_time\_to\_delivery ASC

LIMIT 5

|  |  |  |  |
| --- | --- | --- | --- |
| customer\_state | avg\_freight\_value | avg\_time\_to\_delivery | avg\_diff\_estimated\_delivery |
| AP | 34.00609756 | -28.22222222 | 18.39506173 |
| RR | 42.98442308 | -28.17391304 | 18.32608696 |
| AM | 33.20539394 | -26.33742331 | 19.93251534 |
| AL | 35.84367117 | -24.44730679 | 8.735362998 |
| PA | 35.83268519 | -23.70208729 | 14.25047438 |

Lowest:

SELECT \*

FROM CTE

ORDER BY avg\_time\_to\_delivery DESC

LIMIT 5

|  |  |  |  |
| --- | --- | --- | --- |
| customer\_state | avg\_freight\_value | avg\_time\_to\_delivery | avg\_diff\_estimated\_delivery |
| SP | 15.14727539 | -8.662252654 | 11.20791077 |
| PR | 20.53165157 | -11.89307842 | 13.48610374 |
| MG | 20.63016681 | -11.92072463 | 13.34264922 |
| DF | 21.04135495 | -12.89384289 | 12.20042463 |
| SC | 21.47036877 | -14.95021962 | 11.5727184 |

* + Top 5 states where delivery is really fast/ not so fast compared to estimated date

Fast\_delivery:

SELECT \*

FROM CTE

ORDER BY avg\_diff\_estimated\_delivery ASC

LIMIT 5

|  |  |  |  |
| --- | --- | --- | --- |
| customer\_state | avg\_freight\_value | avg\_time\_to\_delivery | avg\_diff\_estimated\_delivery |
| AC | 40.07336957 | -20.68131868 | 20.97802198 |
| RO | 41.06971223 | -19.65567766 | 20.04029304 |
| AM | 33.20539394 | -26.33742331 | 19.93251534 |
| AP | 34.00609756 | -28.22222222 | 18.39506173 |
| RR | 42.98442308 | -28.17391304 | 18.32608696 |

Not\_so\_fast\_delivery:

SELECT \*

FROM CTE

ORDER BY avg\_diff\_estimated\_delivery DESC

LIMIT 5

|  |  |  |  |
| --- | --- | --- | --- |
| customer\_state | avg\_freight\_value | avg\_time\_to\_delivery | avg\_diff\_estimated\_delivery |
| AL | 35.84367117 | -24.44730679 | 8.735362998 |
| MA | 38.25700243 | -21.59 | 9.90625 |
| SE | 36.65316883 | -21.41866667 | 10.00266667 |
| ES | 22.0587766 | -15.58741573 | 10.64629213 |
| BA | 26.36395894 | -19.19250611 | 10.98262286 |

1. **Payment type analysis**
   * Count of orders for different payment types:

Query:

SELECT pa.payment\_type, COUNT(\*) as order\_count

FROM `scalerproject4.targetsql.payments` as pa

LEFT JOIN `scalerproject4.targetsql.orders` as od

  ON od.order\_id = pa.order\_id

GROUP BY pa.payment\_type

Answer:

|  |  |
| --- | --- |
| payment\_type | order\_count |
| credit\_card | 76795 |
| voucher | 5775 |
| not\_defined | 3 |
| debit\_card | 1529 |
| UPI | 19784 |

* Distribution of payment installments and count of orders:

Query:

SELECT pa.payment\_installments, COUNT(\*) as order\_count

FROM `scalerproject4.targetsql.payments` as pa

LEFT JOIN `scalerproject4.targetsql.orders` as od

  ON od.order\_id = pa.order\_id

GROUP BY pa.payment\_installments

Answer:

|  |  |
| --- | --- |
| payment\_installments | order\_count |
| 0 | 2 |
| 1 | 52546 |
| 2 | 12413 |
| 3 | 10461 |
| 4 | 7098 |
| 5 | 5239 |
| 6 | 3920 |
| 7 | 1626 |
| 8 | 4268 |
| 9 | 644 |
| 10 | 5328 |
| 11 | 23 |
| 12 | 133 |
| 13 | 16 |
| 14 | 15 |
| 15 | 74 |
| 16 | 5 |
| 17 | 8 |
| 18 | 27 |
| 20 | 17 |
| 21 | 3 |
| 22 | 1 |
| 23 | 1 |
| 24 | 18 |

* Count of orders for different payment types Month over Month:

Query:

SELECT EXTRACT(YEAR FROM od.order\_purchase\_timestamp) as years, EXTRACT(MONTH FROM od.order\_purchase\_timestamp) as months, pa.payment\_installments, COUNT(\*) as order\_count

FROM `scalerproject4.targetsql.payments` as pa

LEFT JOIN `scalerproject4.targetsql.orders` as od

  ON od.order\_id = pa.order\_id

GROUP BY years, months, pa.payment\_installments

ORDER BY years, months

LIMIT 10

Answer:

|  |  |  |  |
| --- | --- | --- | --- |
| years | months | payment\_installments | order\_count |
| 2016 | 9 | 2 | 1 |
| 2016 | 9 | 1 | 1 |
| 2016 | 9 | 3 | 1 |
| 2016 | 10 | 1 | 144 |
| 2016 | 10 | 7 | 13 |
| 2016 | 10 | 2 | 30 |
| 2016 | 10 | 3 | 43 |
| 2016 | 10 | 6 | 18 |
| 2016 | 10 | 5 | 20 |
| 2016 | 10 | 4 | 26 |